Re-ranking via User Feedback: Georgetown University at TREC 2015 DD Track

Jiyun Luo and Hui Yang
Department of Computer Science, Georgetown University
37th and O Street NW, Washington DC, USA
jl1749@georgetown.edu, huiyang@cs.georgetown.edu

Abstract

There are two principal components involved in a search process, the user and the search engine. In TREC DD, the user is modeled by a simulator, called "jig". The jig and the search engine exchange many messages among themselves, including the relevant passages returned by the jig, user cost spent on examining the documents, etc. In this work, we don't apply any dynamic search algorithms to model these interactions. Instead, we produce a basic re-ranking baseline. Our algorithm starts at taking in an initial query from the simulator. During the search, we collect the relevance feedback from the simulator and use them to re-rank the initial retrieval results. Our algorithm terminates itself automatically when it senses that the user has gained enough information about the search topic or that no further relevant documents can be retrieved for the user.

1. Introduction

TREC 2015 Dynamic Domain[1] is an interactive search task. There are two components in it, a search engine and a simulating user. The search engine is implemented by the participants, while the simulating user is provided by TREC 2015 DD Track organizer, and is called "jig". There are 118 search topics in total. For each search topic, a short query, which is the name of the search topic, is given to the participant's system at the beginning. Based on the initial query, the participant's system is required to return five documents to the jig. Then the jig provides feedbacks to the retrieval system regarding these five documents. The retrieval system can decide how to utilize these user feedbacks. If the system decides to continue the search procedure, it returns five more documents to the jig, and the jig judges them and returns feedbacks again. This iteration can loop forever and it is the participant system's responsibility to decide when to end the loop. The feedbacks from the jig contain information about topic ID, jig's confidence degree of its judgment, the subtopics that the returned document covers, the relevant passage content, etc. The detail of the feedback is listed in Table 1.

The documents returned at each loop are combined together and judged by TREC DD evaluation metrics. For example, if a search system interacts with the jig for t times and at j^{th} time, it returns a document set D_j, where j belongs to [1, t], then TREC DD combines D_1 to D_t chronologically to form a ranked list and evaluates it. A good retrieval system for TREC DD should return high relevant materials at an early position, should cover as many subtopics as possible, and should minimize users' efforts to examine retrieval results.

[1] http://trec-dd.org/

Our approaches are built upon the Lemur[2] search engine. We use Lemur's default retrieval algorithm to conduct retrieval and then re-rank the retrieved result based on user feedbacks. Our approaches are able to terminate the search process when they sense that the user has gained enough information about the search topic or that no further relevant documents can be retrieved for the user.

Table 1 An Example of the Jig's Feedbacks

```
[
    {
            "topic_id":"DD15-1",
            "confidence":0.987,
            "subtopics":[
                {
                    "subtopic_id": "DD15-1.1",
                    "passage_text": "Federal judge Redden taking himself off the salmon case",
                    "rating": 3,
                }   , {…},
            ],
            "on_topic":1,
    },
    {...}
]
```

The remaining parts of this paper are organized as follow. We discuss each technical approach in detail from Section 2 to Section 4. In Section 5, we present our submissions and the evaluation results. In Section 6, we conclude our work.

2. LM Retrieval Model

For the initial query of each search topic, we don't have any user feedback received yet and the only information we have is the initial query, so we directly use Language Modeling with Dirichlet smoothing [1], which is the default search algorithm in Lemur. For one search term t, the document d's relevance score is calculated as:

$$P(t|q) = \frac{tf(t,d) + \mu P(t|C)}{length(d) + \mu}$$

where length(d) is the length of document d. $tf(t,d)$ is the term frequency of term t in document d. P(t|C) is the probability that the corpus C generates term t in the query. The Dirichlet smoothing parameter μ is set to 5000 empirically.

3. The Feedback Re-ranking Model

[2] http://www.lemurproject.org/

After we retrieve for the initial query using LM, we get a ranked list of documents. Every time when we deliver five documents to the jig, we get some feedbacks from it. Here we use the texts of the relevant passages in the feedbacks to re-rank documents in the original list. The new relevance score of a document d in the original list becomes:

$$P(d|q) = (1-\lambda)P_{old}(d|q) + \lambda * SimScore(d, Feedback)$$

Here $P_{old}(d|q)$ is the original relevance score calculated by LM. *Feedback* is the relevant passage texts collected from the jig's feedbacks. SimScore is a similarity score between the content of document *d* and relevant texts, *Feedback*. We use the cosine similarity to calculate *SimScore*.

4. Stopping Criteria

Our approaches are able to terminate themselves based on user feedbacks. For a search topic, we call the procedure of search engine sending five documents to the jig and the jig returning judgment for the five documents as one iteration. Our approaches terminate themselves if in *n* adjacent iterations no relevant documents are found. We set *n=5*, if not a single relevant document has been found for this topic. Otherwise we set *n=3*. It means that the search engine assumes that in order to finish a search task, the user is willing to spend more effort to examine retrieval results if they find no relevant information yet. Another stopping condition is that for each topic, at most we loop for 20 iterations. We assume that after 20 iterations, the user is tired of current search topic.

5. Experiments

5.1 Domains

We run our experience on three different domains. We index them using Indri. All stopwords are removed and terms are stemmed using Krovetz Stemmer [2].

Ebola. This dataset talks about the 2014-2015 Ebola outbreak in Africa. It contains 497,362 web pages, 19,834 PDFs and 164,961 tweets in total.

Illicit Goods. Documents in this dataset are threads crawled from underground hacking forums, such as BlackHatWorld.com and HackForums.com, which include posts and metadata. It talks about how illicit and fake goods such as fake Viagra are made, advertised, and sold on the Internet. All documents are in the HTML format.

Local Politics. This dataset is related to the regional politics in the Pacific Northwest and the small-town politicians. It is a clean HTML news dataset. This corpus is a subset of the original TREC 2014 KBA Stream Corpus. Table 2 shows the statistics of these three datasets.

Table 2 Statistics Table for TREC DD 2015 Datasets

	Ebola	Local Politics	Illicit Goods
Corpus Size (GB)	12.6	58	8.30
Corpus Size in # of Tokens (10^9)	0.62	1.16	0.21
# of Unique Tokens (10^6)	1.13	1.10	3.55
Avg. Document Length (10^3)	0.95	1.20	0.46

5.2 Submission

GU_RUN3_SIMI. This is a re-ranking approach. In this submission, we use the initial query and the *LM retrieve model* to obtain an original ranked list. Then we use the *Feedback Re-ranking model* to re-rank documents in the original list. We tune the parameter $\lambda = 0.6$.

GU_RUN4_SIMI. This is also a re-ranking approach. It is similar to GU_RUN3_SIMI. The difference is that before we apply the *Feedback Re-ranking model*, we filter the terms in the relevant feedback texts using IDF. Terms with an IDF value less than 0.1 are thrown away.

5.3 Results

Table 3 Evaluation Results for TREC 2015 DD Submissions

	ACT10	CT10	ERRA	ERRH	MPR	PR	RRR	Recall	AP/I	MAP
GU_RUN3	0.104	0.053	0.317	0.237	0.023	0.130	0.490	0.325	0.208	0.226
GU_RUN4	0.105	0.053	0.317	0.236	0.024	0.127	0.490	0.325	0.207	0.225
Median	0.129	0.058	0.264	0.209	0.024	0.129	0.592	0.313	0.225	0.166
Best	0.291	0.265	0.395	0.289	0.162	0.664	1.000	0.646	0.749	0.519

The official evaluation metrics for TREC 15 DD Track are CubeTest [3] and ERR [4]. Table 3 reports the scores of ACT@10, CT@10, ERRA and other metrics for our submissions. It also reports the best and median scores for all submissions in TREC 15 DD Track. The "Best"/ "Median" row doesn't correspond to a real submission. It is constituted by the best/median scores generated by each evaluation metric.

Our submissions are around the median positions for all evaluation metrics. A median Recall value indicates that the original retrieval list used in our re-ranking approaches doesn't contain enough relevant documents. We believe that this is the fact that limits our submissions' performance. Nonetheless in our experiments, we indeed observe CT score improvement in our submitted runs when we compare them with a naive basic approach, where we only apply LM retrieval model for the initial query and ignore all user feedbacks. This indicates that user feedbacks in a search session are informative and hence they shouldn't be discarded.

The GU_RUN3_SIMI and GU_RUN4_SIMI runs have similar scores for all metrics. It suggests that filtering low quality terms in the feedback texts isn't important for re-ranking approaches.

Another observation is that we find out that if we terminate our interaction iterations with the jig earlier, our CT scores will be improved. This indicates that we need a more sensitive stopping strategy than the current one which we describe in Section 4.

After TREC submission deadline, we implement another system where the user feedbacks are not utilized to re-rank an original list, but are used to generate a new query for retrieving at each iteration. The result shows that the new approach doesn't significantly impact the CT score, however it does improve the Recall score. It indicates that the TREC 15 DD Track is a precision sensitive retrieval task.

6. Conclusion

We submit two re-ranking approaches to the TREC 2015 DD Track. Our submissions achieve a median performance among all submissions. Our performance is mainly limited by the low quality of the initial retrieval list used for re-ranking. Based on the evaluation results, we learn that 1) multiple retrievals are needed for TREC 2015 DD Track; 2) the task is precision sensitive, hence a sensitive and sophisticated stopping criteria is necessary; 3) the feedbacks in the search session are informative, and are useful to help improving retrieval accuracy.

7. Acknowledgement

The research is supported by DARPA FA8750-14-2-0226, NSF IIS-1453721, and NSF CNS-1223825. Any opinions, findings, conclusions, or recommendations expressed in this paper are of the authors, and do not necessarily reflect those of the sponsor.

8. References

[1]. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. ACM Trans. Inf. Syst., 22(2):179–214, Apr. 2004.

[2]. Jivani, Anjali Ganesh. "A comparative study of stemming algorithms." Int. J. Comp. Tech. Appl 2.6 (2011): 1930-1938.

[3] Luo, J., Wing, C., Yang, H., Hearst, M. The water filling model and the cube test: multi-dimensional evaluation for professional search. CIKM 2013. Burlingame, CA.

[4] Chapelle, O., Metlzer, D., Zhang, Y., Grinspan, P. Expected reciprocal rank for graded relevance. CIKM 2009. Hong Kong, China.